Event-Detecting Multi-Agent MDPs: Complexity and Constant-Factor Approximation

Akshat Kumar
Department of Computer Science
University of Massachusetts, Amherst
akshat@cs.umass.edu

Shlomo Zilberstein
Department of Computer Science
University of Massachusetts, Amherst
shlomo@cs.umass.edu

Abstract

Planning under uncertainty for multiple agents has grown rapidly with the development of formal models such as multi-agent MDPs and decentralized MDPs. But despite their richness, the applicability of these models remains limited due to their computational complexity. We present the class of *event-detecting* multi-agent MDPs (eMMDPs), designed to detect multiple mobile targets by a team of sensor agents. We show that eMMDPs are NP-Hard and present a scalable 2-approximation algorithm for solving them using matroid theory and constraint optimization. The complexity of the algorithm is linear in the state-space and number of agents, quadratic in the horizon, and exponential only in a small parameter that depends on the interaction among the agents. Despite the worst-case approximation ratio of 2, experimental results show that the algorithm produces *near-optimal* policies for a range of test problems.

1 Introduction

Planning under uncertainty for a team of agents has seen much progress thanks to the development of such models as multi-agent MDPs (MMDPs) [Boutilier, 1999] and decentralized MDPs [Bernstein *et al.*, 2002]. Despite rapid progress, the practical application of these models remains limited due to their computational complexity. MMDPs are conceptually simple, but their complexity is exponential in the number of agents. For DEC-MDPs and POMDPs, optimal algorithms have been shown to be NEXP-Complete even for two agents [Bernstein *et al.*, 2002].

To overcome this computational barrier, research has focussed on domains that exhibit certain structure, such as transition independence [Becker, 2006], structured interaction among agents and decomposable reward functions [Guestrin *et al.*, 2001; Nair *et al.*, 2005]. Sensor networks have emerged as a particularly relevant application domain where coordination among sensors is required to track or detect multiple stochastically moving targets over some finite horizon [Lesser *et al.*, 2003]. Sensors often interact only through a joint reward function and they exhibit *locality of interaction*–each sensor interacts with a *limited* number of neighboring sensors. Significant scalability has been demonstrated for noisy sensors with partial observability using the ND-POMDP model [Nair *et al.*, 2005; Kumar and Zilberstein, 2009], a restricted class of DEC-POMDPs.

Our proposed model, *event-detecting* multi-agent MDP (eMMDP), is a specialized framework for early outbreak detection of stochastically evolving events or targets in a distributed sensor network. A key feature of the model is that detection requires collaboration among agents, but even with perfect collaboration agents may fail to detect an event with certain probability. In addition, the network is resource constrained, not allowing all the events to be detected simultaneously. Agents receive a joint reward for successfully detecting an event and their goal is to maximize the accumulated reward over a finite horizon. Rewards for an event have a certain structure which is detailed below.

The eMMDP model is motivated by several real-world applications. One example is a sensor network application, Distributed Collaborative Adaptive Sensing (DCAS) [Pepyne *et al.*, 2008], where a network of radars collaboratively sense the earth's atmosphere for weather phenomena, such as storms and tornadoes [Manfredi and Kurose, 2007]. Events in DCAS are the stochastically moving storms and *detection* implies extracting the storm's physical attributes such as velocity or moisture content. Coordination among radars is required for this task, and the overall goal is to find the policy (sector scan strategy) for radars such that storms are detected as early as possible, providing critical information about their attributes. A similar problem is that of containing virus propagation in a water distribution network [Ostfeld *et al.*, 2006; Leskovec *et al.*, 2007]. In such water sensor networks, the detection involves containing the spread of viruses as soon as possible to minimize the size of affected population. Several other physical phenomena share these early detection/containment characteristics such as the spread of fire in a building or in wilderness. The goal of early detection makes the reward structure inherently non-increasing. That is, the reward for detection *does not* increase with the horizon. We exploit this reward structure to design an efficient algorithm for eMMDPs with a guaranteed approximation bound.

A key contribution of our work lies in establishing connections between decision-theoretic planning for multi-agent systems, algorithmic concepts such as matroids and submodularity, and constraint optimization. The algorithm we

present for solving eMMDPs, *locally greedy submodular maximization* (LGM), uses these key concepts: a *matroid* representation of the policy search space, *submodularity* of the policy value function, and *constraint optimization* to exploit the locality of interaction and avoid exponential complexity in the number of agents (which MMDPs suffer from). LGM is based on a generic approximation algorithm for submodular function maximization [Fisher *et al.*, 1978]. Unlike other heuristics, it provides strong theoretical guarantee of a factor-2 approximation, which is the worst case ratio between the optimal value and the value produced by the algorithm.

Using submodularity, we also provide a much tighter online bound for *any* eMMDP algorithm, and use it to show that policies produced by LGM are provably near-optimal on a number of sensor network configurations. Furthermore, matroids and submodularity are very general concepts not necessarily limited to eMMDPs. Thus, the ideas developed in this paper can become useful in other practical applications too.

2 Event-detecting MMDPs

An eMMDP is a tuple $\langle \mathcal{A}, L, \mathcal{T}, S, P, A, \{R_i^t\}, d \rangle$, where

- $\mathcal{A}$ is a set of n stationary sensor agents (analogous to radars)
- L is a set of locations
- $\mathcal{T}$ is a set of k targets (storms) which move stochastically between the L locations
- S is a system state space: $S = \times_{i=1}^{k} S_i$, where $S_i \subseteq L$ is the set of locations where target i can be. It must be noted that we use the term *detection* in a richer sense than merely modeling the location of targets, which state-space represents. In many applications, such as DCAS, state can be easily tracked using a specialized tracker application [Manfredi and Kurose, 2007].
- P is a state transition probability: if $s = \langle s_1, \ldots, s_k \rangle$ is a joint state, $P(s'|s) = \prod_{i=1}^{k} P(s_i'|s_i)$, assuming transition independence of targets
- A is a joint action space: $A = \times_{i=1}^{n} A_i$, where A_i is the set of actions for agent i. An agent's action can be to scan a location (i.e. $A_i \subseteq L$) or to idle using the nop action
- $\{R_i^t\} = \{\{R_1^t\}, \ldots, \{R_k^t\}\}$ is a set of reward functions for k targets. R_i^t is the reward for detecting target i at time step t. The reward is nonincreasing with time, that is $\forall i : R_i^1 \geq R_i^2 \geq \ldots \geq R_i^T$, indicating that targets should be detected as early as possible to maximize the reward.

We define the following interaction hypergraph, $G = (V, E)$, to describe the interactions among the agents. There is a node for each agent i. There is a hyper-edge for each location l, which connects the nodes of all the agents that can scan l. Let $\mathcal{A}(l) = \{i | l \in A_i\}$ denote that set of agents for location l. To detect a target present in l, any subset of d agents from $\mathcal{A}(l)$ must scan it. A successful detection produces the associated reward, otherwise the reward is 0. While d can be arbitrary, we assume for simplicity that it is the same for all targets. Clearly, coordination among agents plays an important role in solving eMMDPs. The next section describes the policy structure and its value function.

2.1 Policy structure and value function

At each state s^t, the choice of targets to detect must come from the power set $2^{\mathcal{T}}$. The joint policy π maps each state $s^t \in S \times T$ to a set $\tau \in 2^{\mathcal{T}}$ of targets to detect, i.e., $\pi : S \times T \rightarrow 2^{\mathcal{T}}$. Hiding the actual state-action mapping for agents under this representation helps provide insights about the problem and becomes useful in proving theoretical results later. In section 3.5, we describe how to extract the underlying policy (state-action mapping) and deal with the exponential size of the power set.

Since targets are independent, the policy and its value can be decomposed over the k targets: $V(\pi) = \sum_{i=1}^{k} V(\pi_i)$, where $\pi_i : S \times T \rightarrow \{0, 1\}$. If target i is detectable by the joint policy, i.e., $i \in \pi(s^t)$, then $\pi_i(s^t) = 1$, else 0. We define each $V(\pi_i)$ as follows. Let $p_i^t(\pi_i)$ be the probability of detecting target i at time t according to π_i. It is given by:

$$p_i^t(\pi_i) = \sum_{s \in S} \pi_i(s^t) \cdot P(s^t)$$

The probability $P(s^t)$ of being in state s at time t can be calculated easily using the given transition function and is conditioned on the initial state or belief. Further, if each detection is successful only with a certain probability, $P(success)$, then each term of the above summation is multiplied by it. We use a shorthand of p_i^t for $p_i^t(\pi_i)$ and define $V(\pi_i)$ as follows:

$$\begin{aligned} V(\pi_i) &= p_i^1 R_i^1 + (1 - p_i^1) p_i^2 R_i^2 + \ldots \\ &\quad \ldots + (1 - p_i^1) \cdots (1 - p_i^{T-1}) p_i^T R_i^T \\ &= \sum_{t=1}^{T} \prod_{t'=1}^{t-1} (1 - p_i^{t'}) \cdot p_i^t \cdot R_i^t \end{aligned}$$

Let q_i^t be the product $\prod_{t'=1}^{t-1} (1 - p_i^{t'})$ with $q_i^1 = 1$. This product denotes the probability that target i is *not detected* before time t. This notation further simplifies the value function:

$$V(\pi_i) = \sum_{t=1}^{T} q_i^t \cdot p_i^t \cdot R_i^t.$$

The goal of solving an eMMDP is to find the joint policy π with the highest expected value $V(\pi)$.

2.2 Hardness of eMMDPs

Theorem 1. *Solving an eMMDP is NP-Hard.*

Proof. We reduce the NP-Complete *weighted k-Set Packing* problem to eMMDP. Given a collection $X_1, \ldots, X_q$ of sets of cardinality at most k, with real weights for each set, the k-set packing problem is that of finding a collection of pairwise disjoint sets of maximum total weight. The reduction efficiently maps it to the tuple $\langle L, \mathcal{A}, \mathcal{T}, A, \{R_i\}, d \rangle$.

The set L include q locations, one for each X_i. Let $X = \cup_{i=1}^{q} X_i$. The set $\mathcal{A}$ includes one agent for each element x of X. The action set A_x for agent $\mathcal{A}_x$ includes actions to scan all locations l_{X_i} such that $x \in X_i$, i.e. $A_x = \{l_{X_i} : x \in X_i\}$. One target is present in each location l_{X_i} in the initial state, with associated reward equal to the weight of the set X_i. To detect the target in location l_{X_i}, $|X_i|$ surrounding agents need

to scan it. The time horizon for the problem is 1. Clearly, this reduction is polynomial time. The k in k-set packing signifies that for detecting a target, a fixed number of agents are required (parameter d in eMMDP). Finally we need to show that there exists a collection of disjoint sets with total weight W for the given k-set packing problem if and only if the corresponding eMMDP has a policy with value W.

($\Rightarrow$) Let $X_{i1}, \ldots, X_{ic}$ be a disjoint collection with weight W. Corresponding to a target in location $l_{X_{ij}}$, assign action $l_{X_{ij}}$ to each agent $\mathcal{A}_x : x \in X_{ij}$. Clearly, $|X_{ij}|$ agents scan location $l_{X_{ij}}$, as required by eMMDP. This policy also prevents conflicts among agents in scanning assignment. That is, each agent scans *at most* one location. To prove this, suppose, $\mathcal{A}_x$ has to scan two locations $l_{X_{ij}}$ and $l_{X_{ij'}}$. By the construction of the action set for $\mathcal{A}_x$, to scan locations $l_{X_{ij}}$ and $l_{X_{ij'}}$, both X_{ij} and $X_{ij'}$ must contain x (i.e., $X_{ij} \cap X_{ij'} \neq \phi$). But, this is a contradiction because X_{ij}, $X_{ij'}$ are disjoint. The value of the policy is W because the detected targets have the same rewards as the weights of their corresponding sets.

($\Leftarrow$) Let the detected targets be in locations $l_{X_{i1}}, \ldots, l_{X_{ic}}$, and the policy value be W. Then the sets in the packing problem are $X_{i1}, \ldots, X_{ic}$. Their total weight is W and they are disjoint. Otherwise, for any shared element $x \in X_{ij} \cap X_{ij'}$, $\mathcal{A}_x$ has to scan both locations $l_{X_{ij}}$ and $l_{X_{ij'}}$ to detect the respective targets, which is impossible. □

3 Matroids and eMMDPs

Let E be a finite ground set and $\mathcal{F}$ be a non-empty collection of subsets of E. The system $\mathcal{M} = (E, \mathcal{F})$ is a matroid if is satisfies the following two properties.

- The *hereditary* property: $F_1 \in \mathcal{F} \wedge F_2 \subset F_1 \Rightarrow F_2 \in \mathcal{F}$. In other words, all the subsets of F_1 must be in $\mathcal{F}$.
- The *exchange* property: $\forall F_1, F_2 \in \mathcal{F}$: $|F_1| < |F_2| \Rightarrow \exists x \in F_2 \backslash F_1 \; ; \; F_1 \cup \{x\} \in \mathcal{F}$.

The members of $\mathcal{F}$ are also called *independent sets*. Matroids are useful combinatorial structures, an example of which is the graphic matroid, where E is the set of all edges in a graph and $\mathcal{F}$ is the family of all *acyclic* subsets of edges.

A set $F \in \mathcal{F}$ is called *maximal* if there is no $x \in E$ such that $F \cup \{x\} \in \mathcal{F}$. Many combinatorial optimization problems can be modeled as a matroid, with the optimal solution equivalent to finding the maximal, maximum weight independent set. A classical example on the graphic matroid is finding the maximum weight forest (equivalent to the minimum spanning tree when edge weights are subtracted from a large positive constant). Such set is maximal (i.e., adding an edge to a spanning tree creates a cycle) and maximum weight.

For a ground set partitioned as $E = E_1 \cup E_2 \cup \ldots \cup E_k$, the family of subsets $\mathcal{F} = \{F \subseteq E : \forall i; |F \cap E_i| \leq 1\}$ forms a matroid called a *partition matroid*. This family simply denotes that any independent set can include at most one element from each ground set partition. We show in the following sections that any eMMDP can be mapped to a partition matroid, where the collection $\mathcal{F}$ denotes the policy search space of the eMMDP. The set evaluation function $z : 2^E \rightarrow \Re$ for this matroid is similar to the eMMDP policy evaluation function. Finding an optimal policy for an eMMDP is equivalent to finding the maximal, maximum weight independent set under z.

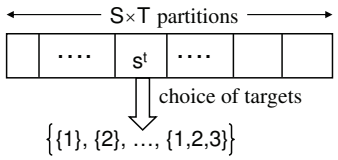


Figure 1: A partitioned view of an eMMDP policy

3.1 eMMDP as a partition matroid

The policy for an eMMDP can be viewed as a collection of $|S| \times T$ partitions. Each partition denotes a unique state-time pair s^t. The policy, $\pi : S \times T \rightarrow 2^{\mathcal{T}}$, constrains each partition to make sure that it is filled by exactly one element from the power set. Fig. 1 illustrates this with target set $\mathcal{T} = \{1, 2, 3\}$. This notion is formalized below as a partitioned ground set.

Let $E = \cup_{s^t} E_{s^t}$, where $E_{s^t} = \{(s^t, \tau) : \tau \in 2^{\mathcal{T}}\}$, be a ground set. The subset system $\mathcal{F} = \{F \subseteq E : \forall s^t; |F \cap E_{s^t}| \leq 1\}$ forms a partition matroid. A member $(s^t, \tau) \in E$ denotes that the policy in state s^t is to detect the targets in the set τ. The policy constraint is also modeled by the partition matroid. Clearly, the independent sets are the *partial policies* for eMMDP and the *maximal* independent set specifies the complete policy. The maximum weight, maximal set represents the optimal eMMDP policy. The weight criterion z, intuitively enough, is the evaluation function of the policy an independent set represents.

3.2 Set evaluation function

The evaluation function $z : 2^E \rightarrow \Re$ is defined for *any* subset of E. For the independent sets, z is the same as the evaluation function of the policy it represents. For partial policies, we assume no targets are detected in unspecified partitions.

Sets F which violate the matroid constraint, i.e., $\exists E_{s^t} : |F \cap E_{s^t}| > 1$, do not represent a valid policy as they include more than one element from a ground set partition. The function π then becomes a *relation*. For such invalid policies, the only change occurs in specifying the policy π_i for each target i, with

$$\pi_i(s^t) = \begin{cases} 1 \text{ if } \exists \tau \in 2^{\mathcal{T}} \; s.t. \; \pi(s^t, \tau) \wedge i \in \tau \\ 0 \end{cases}$$

The detection probability for each target and the final value can be calculated using this modified policy as in Section 2.1.

3.3 Monotonicity and submodularity

If the evaluation function z is linear in the weight of each element, then the greedy approach, which starts with an empty set and chooses the next best element, is *optimal*; this is exactly how Kruskal's algorithm for MST works. However, for eMMDPs, we show that z is *submodular* and *monotone* instead of linear. Surprisingly, a similar greedy approach gives factor-2 approximation [Fisher *et al.*, 1978]. Goundan and Schulz [2007] provide a good overview of these concepts, which we summarize below. A set function z is said to be

- *monotone*, if $z(F_1) \leq z(F_2)$ whenever $F_1 \subseteq F_2$

- *submodular*, if $z(F_1 \cup \{e\}) - z(F_1) \leq z(F_2 \cup \{e\}) - z(F_2)$ for all $F_2 \subseteq F_1 \subseteq E$ and $e \in E \setminus F_1$

Submodularity is similar to the *law of diminishing returns*. As a set grows, the marginal gain of each subsequent element decreases. The *marginal gain* of an element $e \in E \setminus F$ to F is $\rho_e(F) = z(F \cup \{e\}) - z(F)$. Equivalently, submodularity implies $\rho_e(F_1) \leq \rho_e(F_2), \forall F_2 \subseteq F_1$.

For eMMDPs, this implies that extending a smaller partial policy leads to a higher increase in reward than extending its superset. Intuitively, in a smaller policy there are *at least* as many undetected targets as in its superset, possibly more. Therefore, the chance that a new target will be detected providing extra reward is higher when the smaller policy is extended.

Theorem 2. *The evaluation function for an eMMDP is monotone and submodular.*

Proof. For monotonicity, it suffices to show $\rho_e(F_1) \geq 0$. For eMMDPs, adding $e = (s^t, \tau)$ can never *decrease* the probability of detecting *any* target, thus, $\rho_e(F_1) \geq 0$.

Consider the sets $F_1 \subseteq F_2 \subseteq E$, and an element $e = (s^{t+1}, \tau) \in E \setminus F_2$. Intuitively, we are extending the partial policy F_2 by specifying targets to detect in state s^{t+1}. Given, $V(\pi) = \sum_{i=1}^{k} V(\pi_i)$, we will show that each $V(\pi_i)$ is submodular; submodularity of the joint policy follows by the property that a positive linear combination of submodular functions is also submodular.

With $\pi_i^{F_2}$ we represent the underlying policy of set F_2. $V(\pi_i^{F_2}) = \sum_{t=1}^{T} q_i^t \cdot p_i^t \cdot R_i^t$. We can represent the above summation as a sum of values obtained from time step 1 to t ($= v_1^t$), the value at step $t+1$, and the remaining value from step $t+2$ to the end (v_{t+2}^T). Thus,

$$V(\pi_i^{F_2}) = v_1^t + q_i^{t+1} p_i^{t+1} R_i^{t+1} + q_i^{t+1}(1 - p_i^{t+1}) v_{t+2}^T \quad (1)$$

Consider the set $F_2 \cup \{e = (s^{t+1}, \tau)\}$. If target $i \notin \tau$, then $\rho_e(F_2) = 0$, and proving submodularity is trivial as $\rho_e(F_1) \geq 0 = \rho_e(F_2)$ by monotonicity. Thus, we consider the nontrivial case when $i \in \tau$. In Eq. (1), only the middle and the last term will change by including e. Specifically, the probability of detecting target i at time $t+1$ will increase by $P(s^{t+1})$, the probability of state s at time $t+1$, which can be calculated easily from the input parameters. Thus, we have

$$V(\pi_i^{F_2} \cup \{e\}) = v_1^t + q_i^{t+1}(p_i^{t+1} + P(s^{t+1})) R_i^{t+1} + q_i^{t+1}(1 - p_i^{t+1} - P(s^{t+1})) v_{t+2}^T \quad (2)$$

$\rho_e(F_2)$ is given by subtracting Eq. (1) from Eq. (2):

$$\rho_e(F_2) = q_i^{t+1} P(s^{t+1}) R_i^{t+1} - q_i^{t+1} P(s^{t+1}) v_{t+2}^T \quad (3)$$

Similarly, $\rho_e(F_1)$ is given by

$$\rho_e(F_1) = q_i^{t+1} P(s^{t+1}) R_i^{t+1} - q_i^{t+1} P(s^{t+1}) v_{t+2}^T \quad (4)$$

To show submodularity, we must prove $\rho_e(F_1) - \rho_e(F_2) \geq 0$. Since, F_1 is a subset of F_2, the value of the sub-policy from time step $t+2$ until T in $\pi_i^{F_2}$ cannot be less than $\pi_i^{F_1}$, that is $v_{t+2}^T(\pi_i^{F_2}) \geq v_{t+2}^T(\pi_i^{F_1})$. To get an upper bound on $\rho_e(F_2)$, we substitute $v_{t+2}^T(\pi_i^{F_2}) = v_{t+2}^T(\pi_i^{F_1})$. We get

$$\rho_e(F_1) - \rho_e(F_2) = P(s^{t+1}) \cdot (q_i^{t+1}(\pi_i^{F_1}) - q_i^{t+1}(\pi_i^{F_2})) \cdot (R_i^{t+1} - v_{t+2}^T) \quad (5)$$

Algorithm 1: Locally Greedy Submodular Maximization

```
A^G ← φ
for s^t ∈ S × T do
    e* ← arg max_{e ∈ E_{s^t}} ρ_e(A^G)
    A^G ← A^G ∪ e*
return A^G
```

Since, $\pi_i^{F_1}$ is a subpolicy of $\pi_i^{F_2}$, the probability of *not detecting* target i is higher in $\pi_i^{F_1}$ than $\pi_i^{F_2}$, i.e. $q_i^{t+1}(\pi_i^{F_1}) \geq q_i^{t+1}(\pi_i^{F_2})$. Also, from the reward dependence on the time horizon we have $v_{t+2}^T \leq R_i^{t+2} \leq R_i^{t+1}$. Thus, all the terms in Eq. (5) are ≥ 0. Hence, $\rho_e(F_1) - \rho_e(F_2) \geq 0$, implying submodularity. Note that these results hold even when F_2 represents a non-valid policy. The proof is similar, but omitted due to space constraints. □

3.4 A submodular greedy algorithm

This section describes the 2-approximation greedy algorithm, LGM, for solving eMMDPs. It is based on an approach for maximizing submodular functions over a partition matroid. For details and proof of approximation quality guarantee we refer to Fisher *et al.* [1978] and a recent survey by Goundan and Schulz [2007]. The worst-case bound may seem loose, but using submodularity, we also provide a way to calculate much tighter online bounds, which we discuss later.

The surprisingly simple and efficient LGM technique, detailed in Algorithm 1 above, iteratively builds the policy A^G. Starting with an empty set, the algorithm iterates over the policy partitions (as Fig. 1 depicts). In each iteration, it selects the element from the current partition which maximizes the marginal gain. The ordering of the partitions does not affect the quality guarantee, though a good ordering may provide a better solution.

It should be noted that LGM is *not* a myopic algorithm in the sense that it does take into account the sequential nature of the eMMDP policy. During the initial iterations, when A^G is relatively sparsely filled, it has some myopic nature, but as the policy A^G gets built progressively, the marginal gain step will take into account which targets already have high detection probability and focus the next iteration on targets with low current detection probability. This step combined with the submodularity of eMMDPs is the key to the strong quality guarantees provided by LGM. Other myopic heuristics–such as detecting targets that provide maximum *immediate* reward in each state using constraint optimization–cannot provide such quality guarantees.

Complexity

The complexity of LGM depends upon the number of partitions and the complexity of selecting the best element $e^\star$ from a partition. The number of partitions is $|S| \cdot T$, linear in the state-space and horizon. However, maximizing the marginal gain requires iterating over $2^{|\mathcal{T}|}$ alternatives, which is exponential in the number of targets. To address this, we transform the problem below into a constraint optimization problem,

which can be solved much faster in $O(n \cdot |A_i|^m)$ time and space using the bucket elimination approach [Dechter, 1999]. A_i is the action set for an agent, n is the number of agents, and m is the *induced width* of the chosen depth-first search (DFS) ordering of the variables in the constraint formulation. The constraint graph we construct is similar to the agent interaction graph defined earlier–a hyper-edge can be simulated using an n-ary constraint involving all agents on the edge. Such formulation makes the complexity *linear* in the number of agents, a significant contribution that increases the scalability of the approach to large multi-agent systems. In loosely-coupled systems, the induced width is low, making the constraint formulation particularly easy to solve. Setting up the constraint formulation requires calculating marginal gain of each target, which can be calculated in $O(T)$ time. Hence, the overall complexity of LGM is $O(|S|T(|\mathcal{T}|T+n|A_i|^m)))$.

3.5 Maximizing the marginal gain

Before presenting the constraint formulation, we prove the following theorem. Intuitively, it states that the marginal gain for detecting a group of j targets in state s^t can be factored and represented as a sum of gains provided by *each* target.

Theorem 3. *Let $F \subseteq E$ and $e = (s^t, \tau)$ is an element of $E \setminus F$, where $\tau = \{i_1, \ldots, i_j\}$ represents the choice of j targets to detect in state s^t. Then, $\rho_e(F) = \sum_{c=1}^{j} \rho_{e_c}(F)$, where $e_c = (s^t, \{i_c\})$.*

Proof. First note that $\rho_e(F) = z(F \cup (s^t, \tau)) - z(F)$, which can be written as a telescoping series with $\tau_c = \{i_1, \ldots, i_c\}$

$$\rho_e(F) = \sum_{c=1}^{j} \{\, z(F \cup (s^t, \tau_c)) - z(F \cup (s^t, \tau_{c-1})) \,\} \quad (6)$$

Each term on the RHS of Eq. (6) can be further decomposed with respect to the targets $c' \in \mathcal{T}$ which do not occur in τ_c; for them the policy valuation remains unchanged, and the targets in τ_c; for which the valuation changes.

$$\begin{aligned} z(F \cup (s^t, \tau_c)) &= \sum_{c' \in \mathcal{T} \setminus \tau_c} V(\pi^F_{c'}) + \sum_{c' \in \tau_c} V(\pi^F_{c'} \cup e_{c'}) \\ z(F \cup (s^t, \tau_{c-1})) &= \sum_{c' \in \mathcal{T} \setminus \tau_{c-1}} V(\pi^F_{c'}) + \sum_{c' \in \tau_{c-1}} V(\pi^F_{c'} \cup e_{c'}) \end{aligned}$$

Subtracting the latter equation from the previous one gives

$$\begin{aligned} z(F \cup (s^t, \tau_c)) - z(F \cup (s^t, \tau_{c-1})) &= \\ -V(\pi^F_c) + V(\pi^F_c \cup e_c) &= \rho_{e_c}(F) \end{aligned}$$

The theorem follows immediately by substituting this result with the right hand side of Eq. (6). □

We conclude from Theorem 3 that if $e = (s^t, \{i_1, \ldots, i_j\})$, then maximizing the marginal gain is equivalent to finding $e^\star \leftarrow \arg\max_e \sum_{c=1}^{j} \rho_{e_c}(A^G)$. Constraint optimization algorithms can efficiently solve this problem. Further, constraint optimization provides the actual policy (state-action mapping) for each agent in addition to maximizing marginal gain.

The constraint formulation

A constraint optimization problem (COP) is defined by a set of variables $\mathcal{X} = \{X_1, ..., X_n\}$ and a set of constraint functions $\mathcal{F} = \{f_1, ..., f_m\}$. Each variable X_i has a domain D_i of possible values. Each constraint f_j involves some subset of variables and specifies a *reward* for each value of the subset. The solution is the complete assignment $\mathcal{X}^*$ for the variables that *maximizes* the global reward.

Variables and domain: For each target $i \in \mathcal{T}$, let $l_i \in L$ be its location in state s^t. $\mathcal{A}(l_i)$ denotes the agents which can scan l_i. The constraint network includes a variable for each agent in the group $\mathcal{A}(l_i)$ (and for all targets i), but without any duplicates. The *domain* of each variable is the action set of the corresponding agent.

Constraints: To detect a target i and claim the reward, d agents from $\mathcal{A}(l_i)$ must scan l_i. To ensure this coordination, an n-ary constraint f_i is created among all the variables, say $X(l_i)$, corresponding to agents $\mathcal{A}(l_i)$ (and for all targets i). The valuation for the joint assignment x to $X(l_i)$ is defined below. X_d denotes a group of d variables and x_X is the projection of assignment x on variable X.

$$f_i(x) = \begin{cases} \rho_{(s^t, \{i\})}(A^G) \text{ if } \exists X_d \subseteq X(l_i) : \forall X \in X_d \; x_X = l_i \\ 0 \text{ otherwise} \end{cases}$$

The above constraint valuation ensures that reward is only given when at least d agents from $\mathcal{A}(l_i)$ scan location l_i. This formulation maximizes $\sum_{i=1}^{k} f_i$, where k is the number of targets. Clearly, the optimal solution includes detecting those targets which maximize this sum, which is also the *maximum marginal gain*. The group of detected targets can be extracted from solution $\mathcal{X}^*$ by examining each f_i, and including target i in $e^\star$ for which $f_i \neq 0$ under $\mathcal{X}^*$. Furthermore, the solution $\mathcal{X}^*$ denotes which action each agent must take, hence provides the *actual policy* for agents at state s^t. If an agent is not included in $\mathcal{X}^*$, then it takes the nop action.

3.6 Online bounds

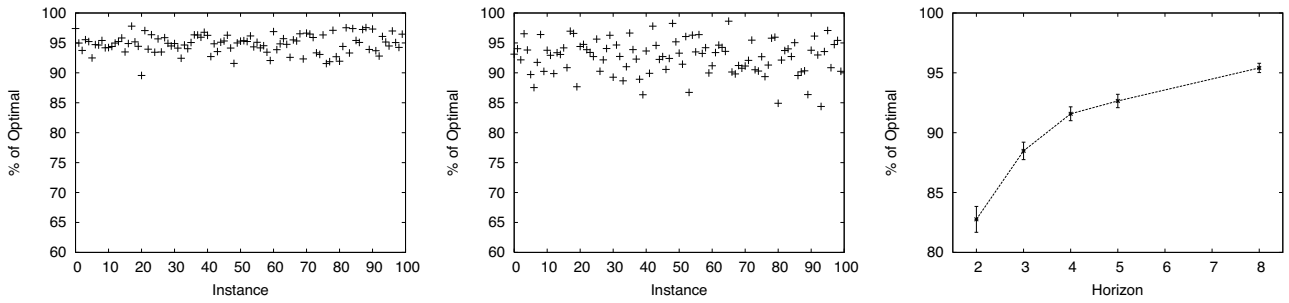
Theorem 4. *Let A^G be a policy returned by some arbitrary eMMDP algorithm, and $A^\star$ be the optimal policy. Then, $z(A^\star) \leq z(A^G) + \sum_{E_{s^t}} \rho^\star_{E_{s^t}}(A^G)$, where $\rho^\star_{E_{s^t}}(A^G) = \max_{e \in E_{s^t}} \rho_e(A^G)$ is the maximum marginal gain any element of the partition E_{s^t} can provide.*

Proof. The proof is based on the following property of submodular functions. If S and T are any subsets of the ground set E, then $z(T) \leq z(S) + \sum_{e \in T-S} \rho_e(S)$. If we substitute $T = A^\star$ and $S = A^G$, then we get

$$z(A^\star) \leq z(A^G) + \sum_{e \in A^\star - A^G} \rho_e(A^G).$$

The optimal solution $A^\star$ is unknown, but we know it must respect the eMMDP partition matroid constraint, i.e., it must pick only 1 element from each partition. Thus, $\sum_{e \in A^\star - A^G} \rho_e(A^G) \leq \sum_{E_{s^t}} \rho^\star_{E_{s^t}}(A^G)$. From this, the theorem follows. □

This property provides tight bounds for *any* eMMDP algorithm and, as the experiments show, is quite useful in practice.



(a) (b) (c)

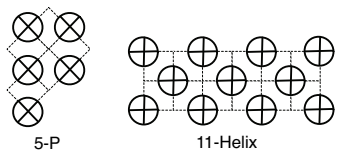
Figure 2: Solution quality for (a) 5P and (b) 11-Helix, for horizon 5. (c) Dependence of solution quality on horizon for 11-helix.

4 Experiments

We performed experiments on multiple sensor network configurations using a 2.4GHz dual-core Mac, with 512MB allotted to JVM. We used a publicly available constraint solver for implementing the marginal gain maximization step of LGM [Petcu and Faltings, 2005].

Fig. 3 shows the first two configurations from [Nair *et al.*, 2005]. Each sensor can scan in four directions. Every square shaped region between sensors represents a location where a target can be. To detect a target, two sensors must scan the location. For the 5P domain, we had three stochastically moving targets. A target can appear in any location with some probability (chosen randomly for each simulation), moves to the next location with probability 0.8 and remains at the current location with probability 0.2 (as in [Nair *et al.*, 2005]). Clearly, not all targets can be detected at the same time: only one among the two targets in the same vertical column can be detected. For each simulation, rewards were selected from the range $[50, 200]$. We generated 100 random instance with horizon 5. Fig. 2(a) shows the quality of approximation for each instance. Clearly, the algorithm proves very effective in this case, producing near-optimal results for all input instances. The average quality achieved was $94.8 \pm 0.3\%$ of the optimal (the error term denotes 95% confidence interval). Average runtime was 53 ms.

Fig. 2(b) shows the approximation ratios for the larger 11-Helix configuration with horizon 5. There were 6 targets, with a state-space of 729 and action-space of 4^{11}. Again, the algorithm was able to produce near-optimal results for all the 100 random problem instances. Average quality was $92 \pm 0.6\%$ and average runtime was 426 ms. Despite an action-space of 4^{11}, our results show that the constraint formulation made it possible to solve the problem efficiently by exploiting the *locality of interactions* among the agents, and



(a) (b)

Figure 3: 5P and 11-Helix sensor network configurations

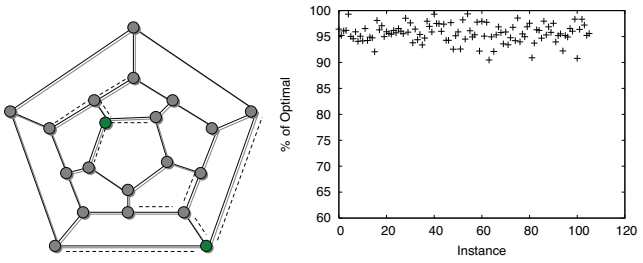
Horizon	2	3	4	5	8
Time (ms)	163	247	334	426	717

Table 1: Horizon versus runtime for 11-Helix

LGM was able to produce near-optimal solutions by exploiting the submodularity of eMMDPs.

Fig. 2(c) shows how solution quality varies with the horizon. Each result is an average over 100 random instances, shown along with 95% confidence intervals. The graph clearly reflects the submodular nature of eMMDP policies. For the smaller horizons, the error term in the formulation of Theorem 4, $\sum_{E_{s^t}} \rho^{\star}_{E_{s^t}}(A^G)$, is higher as the policy size is small. The error decreases as the policy size increases with the horizon–exhibiting the property of diminishing returns. Nevertheless, we were able to achieve very high quality for each horizon. Table 1 shows the runtime for each horizon. The runtime clearly increases polynomially with the horizon, confirming the good scalability of our approach.

For the next set of experiments, we used a much larger configuration–a dodecahedron with 20 nodes and 30 edges (or locations for targets). There were 7 stochastically moving targets and two adjacent agents were required to detect a target. The configuration and sample trajectories are shown in Fig. 4(a). In this configuration, 4 targets can appear in the innermost pentagon with randomly chosen probability and 3 in the outermost. The edges adjacent to the top left innermost node (shown in green) are the possible initial locations of a target. Targets can move stochastically to an adjacent edge, or jump to an edge on the outer pentagon with randomly chosen probabilities. With 0.2 probability targets remain at a location connecting two regions. The rewards were chose randomly for each instance. Similar trajectories were defined for each of the 7 targets such that targets in the innermost pentagon stochastically move to an outer pentagon, and targets in outer pentagons move to an inner pentagon. This makes the problem combinatorially complicated, as it presents agents with many alternatives that they must evaluate. The size of the state-space is 78,125 and the actions-space is 3^{20}. Using reachability analysis, it was possible to reduce the state-space to an average of 16,000 per instance. Fig. 4(b) shows the approximation quality for each of the 100 instances. Again, LGM produced near-optimal results. The average quality was $95 \pm 0.4\%$. The average runtime was



(a) (b)

Figure 4: (a) The larger 20D sensor network configuration, and (b) approximation quality for horizon 5

66 ± 1.7 sec. These results clearly demonstrate the scalability and approximation quality of our approach for large state and action spaces. Because the complexity is linear in the state-space, based on these experiments we could solve a problem with 1 million states in just over an hour, which is a major improvement over the state-of-the-art.

For comparison purposes, we implemented another heuristic approach. In this approach, the group of targets which maximize the immediate joint reward are selected for detection in each state, instead of the ones which maximize the marginal gain. Clearly, this is suboptimal as it completely disregards the sequential and stochastic nature of eMMDPs. For randomly generated instances, as expected, LGM *always* produces better value than this heuristic. The gain in solution quality provided by LGM varies due to the random nature of these experiments. On average, LGM provided 20-30% gain, and in many instances it nearly doubled the heuristic value on the 11-Helix domain.

5 Conclusion

Sensor networks often require coordination of several sensors to detect and identify target phenomena such as weather conditions. As the size of these networks grow, the complexity of coordinating the sensors and the sequential nature of the decision process makes it hard to achieve good performance. We present a formal framework, eMMDP, to study the general problem of outbreak detection and showed that it is NP-Hard. The main contributions of this work are: (1) showing that eMMDPs exhibit *submodularity*, and (2) establishing their connection with *matroids* to provide a simple, principled approximation algorithm for solving them. The approximation algorithm provides strong theoretical guarantee of factor-2 in the worst-case. Using submodularity we also provided much tighter online bound for any eMMDP algorithm and use it to show that LGM produces near-optimal policies.

To improve the scalability of the algorithm, a key step was reformulated as a constraint optimization problem. This made the complexity *linear* in the number of agents and exponential only in the induced tree-width of the interaction graph, which is often much smaller than the number of agents.

Matroids and submodularity are very general concepts that are likely to be useful in other contexts and help develop practical approximation algorithms. For example, any MDP and some classes of decentralized MDPs (such as transition-independent DEC-MDPs) can be easily modeled as a partition matroid. Identifying domains that exhibit submodularity for these classes will allow a straightforward application of LGM and provide efficient approximation with similar strong theoretical guarantees.

Acknowledgments

This work was funded in part by the National Science Foundation under grant IIS-0812149 and by the Air Force Office of Scientific Research under grant FA9550-08-1-0181.

References

[Becker, 2006] R. Becker. *Exploiting Structure in Decentralized Markov Decision Processes*. PhD thesis, University of Massachusetts Amherst, 2006.

[Bernstein *et al.*, 2002] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, 27:819–840, 2002.

[Boutilier, 1999] C. Boutilier. Sequential optimality and coordination in multiagent systems. In *IJCAI*, pages 478–485, 1999.

[Dechter, 1999] R. Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(1-2):41–85, 1999.

[Fisher *et al.*, 1978] M. L. Fisher, G. Nemhauser, and L. A. Wolsey. An analysis of approximations for maximizing submodular set functions-II. *Mathematical Programming Study*, 8:73–87, 1978.

[Goundan and Schulz, 2007] P. R. Goundan and A. S. Schulz. Revisiting the greedy approach to submodular set function maximization. In *Optimization online*, 2007.

[Guestrin *et al.*, 2001] C. Guestrin, D. Koller, and R. Parr. Multiagent planning with factored MDPs. In *NIPS*, pages 1523–1530, 2001.

[Kumar and Zilberstein, 2009] Akshat Kumar and Shlomo Zilberstein. Constraint-based dynamic programming for decentralized POMDPs with structured interactions. In *AAMAS*, 2009.

[Leskovec *et al.*, 2007] Jure Leskovec, Andreas Krause, Christos Faloutsos Carlos Guestrin and, Jeanne VanBriesen, and Natalie Glance. Cost-effective outbreak detection in networks. In *KDD*, pages 420–429, 2007.

[Lesser *et al.*, 2003] V. Lesser, M. Tambe, and C. L. Ortiz, editors. *Distributed Sensor Networks: A Multiagent Perspective*. Kluwer Academic Publishers, Norwell, MA, USA, 2003.

[Manfredi and Kurose, 2007] V. Manfredi and J. Kurose. Scan strategies for adaptive meteorological radars. In *NIPS*, pages 993–1000, 2007.

[Nair *et al.*, 2005] R. Nair, P. Varakantham, M. Tambe, and M. Yokoo. Networked distributed POMDPs: A synthesis of distributed constraint optimization and POMDPs. In *AAAI*, pages 133–139, 2005.

[Ostfeld *et al.*, 2006] A. Ostfeld, J. G. Uber, and E. Salomons. Battle of water sensor networks: : A design challenge for engineers and algorithms. In *WDSA*, 2006.

[Pepyne *et al.*, 2008] D. Pepyne, D. Westbrook, B. Philips, E. Lyons, M. Zink, and J. Kurose. Distributed collaborative adaptive sensor networks for remote sensing applications. In *American Control Conference (ACC)*, pages 4167–4172, 2008.

[Petcu and Faltings, 2005] Adrian Petcu and Boi Faltings. DPOP: A scalable method for multiagent constraint optimization. In *IJCAI*, pages 266–271, 2005.